interconnects. Using this technique, it may be possible to optimize complex instruction set computers to run as fast as reduced instruction set computers.)

The invention would respond to a "jump" (a boolean) instruction by selecting certain ones of the random logic elements and would respond to an "add" (an arithmetic) instruction by selecting another group of the random logic elements, each such selection causing the interconnections among the custom logic elements to be reconfigured.

The foregoing concept appears to be revolutionary in the art. For example, the reference now relied upon in rejecting all of the claims, namely Morton, suggests nothing about selecting groups of individual logic elements (as in the random logic elements on a processor chip). Instead, Morton merely suggests controlling the assignment of each one of a set of uniform processors among bit slices of the data needed to be operated upon. Morton suggests nothing in the way of making such processors operate more efficiently. Applicant's invention would be an improvement within an individual processor. Morton teaches single instruction multiple data processing, which, in the prior art, typically means that all of the processors are uniform. Therefore, Morton would have no reason to attempt to classify an instruction to see which one of the processors would execute it most efficiently, since all of his processors would execute the same instruction with the same efficiency. Therefore, it would make no sense to attempt to classify an instruction. (Furthermore, as will be discussed below, no single processor in Morton executes a given instruction, but rather many (or all) of his processors cooperate together to execute the instruction.)

Morton is directed to an entirely different problem, namely how to best organize bit slices of data operands and data results among uniform bit slice processors. As can best be gleaned from Morton's disclosure, the bit slice widths must be selected appropriately given the arithmetic requirement. For example, multiplication of two 16 bit operands requires that the result have 32 bits at least, requiring either four 8 bit slices or two 16 bit slices, depending upon the capability of the individual processors.

In this, all of Morton's processors cooperate to perform the **same** instruction simultaneously, each processor performing the instruction with respect to a particular one of the bit slices involved. In the applicant's invention, different logic elements perform different instruction, each instruction causing a different selection of logic elements or logic element groups, and the resulting action (from the **instruction** classification) engages heterogenous logic elements. Morton is only concerned with arranging bit slices among individual bit slice processors as homogeneous logic elements so that the appropriate bit widths are accommodated for each arithmetic operation. All of Morton's processors cooperate and execute a given instruction. Different instructions are not allocated among different ones of the processors in Morton. Accordingly, Morton has nothing whatsoever to do with applicant's invention and can not possibly suggest it.

Applicant's claims, for example Claim 1, are specifically directed to a combination of elements, including among other things, elements which classify instructions to be executed in accordance with a function to be performed by the instruction being classified. This feature of the invention was explored in detail in the course of applicant's remarks made in the previous Amendment received in the U.S. Patent and Trademark Office on February 19, 1991.

In the latest office action dated May 21, 1991, the Examiner's remarks appeared to be directed to a different combination having nothing to do with the combination claimed by the applicant. Specifically, the Examiner's remarks concern **classifying the data**, as opposed to classifying the